Revista de Informática Teórica e Aplicada - RITA - ISSN 2175-2745
Vol. 30, Num. 01 (2023) 66-77

RESEARCH ARTICLE

Performance Assessment of a Wireless Mesh Network for Post-harvest Food Quality Traceability of Fruit Products: A Case Study

Avaliação de Desempenho de uma Rede em Malha Sem Fios para Rastreabilidade da Qualidade Alimentar no Pós-colheita de Produtos Frutícolas: Um Caso de Estudo

Tiago Costa[1], Luís Santos[1], João M. L. P. Caldeira[1,2]*, Vasco N. G. J. Soares[1,2], Pedro D. Gaspar[3,4]

Resumo: This paper presents the performance evaluation of a wireless sensor mesh network, for monitoring temperature and humidity in horticultural products when transported in truck galleys. For this purpose a software solution was proposed using ESP8266 devices powered by batteries. The mesh network was managed by the painlessMesh library. The proposed solution aims to minimize the energy consumption of the sensor nodes. The validation of the solution was performed in an area simulating a galley of a truck, where five sensor nodes and a root node were distributed. The tests were developed considering four different models involving variations in messages delivery confirmation, number of attempts until successful delivery and duty cycle duration of the nodes. The performance evaluation of the solution aimed to determine, connectivity rate, sending rate after connection and delivery rates of the first and second attempts. The results obtained show that the message delivery confirmation does not bring added value to the solution, contributing only to increase energy consumption. The use of synchronous duty cycles also showed worse results than the asynchronous use. These results allow the creation of a knowledge base for the use of this solution in a real context.
Keywords: Wireless mesh networks — ESP8266 — PainlessMesh — Performance assessment

Resumo: Este artigo apresenta a avaliação de desempenho de uma rede de sensores sem fios organizada em malha, para monitorização da temperatura e humidade em produtos hortofrutícolas quando transportados em galeras de camiões. Para tal é proposta uma solução de software, usando a biblioteca painlessMesh para gestão da rede em malha, desenvolvida para dispositivos com módulo de comunicação ESP8266 alimentados por baterias. Esta solução tem como objetivo minimizar os gastos energéticos dos nós sensores usados. Para efeitos de validação desta proposta foi usada uma área correspondente à galera de um camião, onde foram distribuídos cinco nós sensores e um nó raiz. Os testes foram desenvolvidos considerando quatro modelos diferentes envolvendo, variações na garantia de entrega das mensagens, o número de tentativas até sucesso de entrega e duração dos tempos de adormecimento dos nós. A avaliação de desempenho da solução teve por objetivo determinar, a taxa de conectividade, a taxa de envio após conexão e as taxas de entrega das primeira e segunda tentativas. Os resultados obtidos evidenciam que a confirmação de entrega de mensagens não traz mais valias à solução, contribuindo apenas para o incremento dos gastos energéticos. O uso de tempos sincronizados no adormecimento dos nós, demonstrou também resultados piores que o uso de tempos assíncronos. Estes resultados permitem criar uma base de conhecimento para a utilização desta solução em contexto real.
Palavras-Chave: Redes em malha sem fios — ESP8266 — PainlessMesh — Avaliação de desempenho

[1] *Instituto Politécnico de Castelo Branco (IPCB), Castelo Branco - Beira Baixa, Portugal*
[2] *Instituto de Telecomunicacoes, Covilha - Beira Baixa, Portugal*
[3] *Center for Mechanical and Aerospace Science and Technologies (C-MAST), Universidade da Beira Interior (UBI), Covilhã - Beira Baixa, Portugal*
[4] *Departamento de Engenharia Eletromecânica, Universidade da Beira Interior (UBI), Covilhã - Beira Baixa, Portugal*
***Corresponding author**: jcaldeira@ipcb.pt
DOI: http://dx.doi.org/10.22456/2175-2745.125567 • **Received:** 30/06/2022 • **Accepted:** 18/01/2023

1. Introdução

O crescente aumento da relevância da Internet das Coisas (do Inglês, *Internet of Things*, IoT) [1] num número alargado de cenários de aplicação (cidades inteligentes, casas inteligentes, segurança inteligente, transporte inteligente, agricultura inteligente, saúde inteligente, entre outros) [2, 3, 4], traz consigo a necessidade do desenvolvimento contínuo de protocolos, redes, dispositivos sensores e sistemas.

As redes em malha sem fios (do inglês, *wireless mesh networks* [5]) assumem um papel determinante no paradigma IoT. São caracterizadas por uma topologia em que os nós se agregam numa malha sem uma hierarquia definida, permitindo que todos os dispositivos possam ser uma parte ativa na transmissão dos pacotes [5]. Esta topologia oferece maior tolerância a falhas, visto que os nós não estão dependentes uns dos outros, e facilita o processo de adicionar novos nós à rede sem a necessidade desta ser alterada ou reconfigurada [6]. Apresentam-se como uma solução cómoda e de custos reduzidos, que permite, por exemplo, recolher informação captada por sensores [7]. Com isso, as redes em malha agilizam e flexibilizam a mobilidade dos dispositivos, reduzindo, e por vezes eliminando, a necessidade de pontos de acesso [8].

Um dos grandes domínios onde a importância da IoT se manifesta é o setor agroalimentar. A IoT tem aqui um papel determinante no aumento da eficácia nos processos, na redução de custos e aumento da produção. Parâmetros como a temperatura e humidade são essenciais em fruticultura para assegurar a conservação no pós-colheita [9]. Estes podem ser complexos de monitorizar especialmente no processo de transporte, dada a mobilidade dos veículos e o reduzido alcance do sinal da transmissão de dados. Topologias de redes convencionais não são capazes de se adaptar às rápidas mudanças das condições que ocorrem neste cenário, traduzindo-se em perdas de informação relevante. Os custos de implementação e manutenção dessas redes são ainda outro ponto negativo. Nestes casos, o uso de redes em malha pode ser um contributo relevante para o alargamento das áreas de cobertura das redes de monitorização.

Enquanto região produtora de cereja e pêssego, duas das culturas frutícolas cujo impacto se destaca na economia local, a Cova da Beira, no interior de Portugal, apresenta-se como um cenário de excelência para a introdução de soluções IoT no contexto da agricultura inteligente. Através de sistemas de recolha autónoma de parâmetros específicos, poderá ser possível assegurar a preservação da qualidade da fruta fresca pré-cortada desde a colheita até ao consumidor final. Estes sistemas podem vir a contribuir para reduzir perdas por desperdício nas fases de armazenamento e transporte dessas frutas. O Projeto PrunusPós [10] tem como um dos seus objetivos a proposta e criação de uma solução que possibilite monitorizar a temperatura e a humidade de contentores de fruta armazenados em câmaras frigoríficas e/ou transportados em camiões frigoríficos ao longo de toda a cadeia logística.

Este trabalho parte das conclusões apresentadas num artigo anterior dos mesmos autores [11], que testou e avaliou o desempenho de dispositivos ESP8266 [12] organizados numa rede em malha sem fios gerida pela biblioteca *painlessMesh* [13]. Esse trabalho permitiu avaliar e validar a utilização deste tipo de redes no contexto do projeto PrunusPós. O trabalho apresentado neste artigo teve como objetivo desenvolver algoritmos que garantissem a correta entrega de parâmetros (temperatura e humidade) recolhidos por cinco nós sensores a um nó raiz de uma rede em malha sem fios, minimizando o impacto na autonomia da bateria desses nós sensores. A avaliação da proposta incidiu sobre o contributo para a solução do uso de mecanismos de garantia ou não da entrega das mensagens enviadas e o efeito do uso de uma temporização síncrona ou assíncrona no período de adormecimento dos nós. Neste contexto, a secção 2 apresenta o conceito de redes em malha sem fios. A secção 3 descreve o cenário de aplicação implementado, ao nível de *hardware*, e algoritmos utilizados. A secção 4 aborda as configurações utilizadas, as métricas de desempenho e a análise e discussão dos resultados obtidos. Por fim, a secção 5 conclui o artigo e apresenta o trabalho futuro.

2. Redes em Malha Sem Fios

As redes em malha sem fios são compostas por diversos nós que podem atuar em simultâneo como Pontos de Acesso e como *host* da rede [14]. Estas redes, embora extremamente flexíveis, têm as suas limitações, nomeadamente na qualidade de serviço (QoS, *Quality of Service*, em inglês) por vezes instável que pode levar a períodos de elevada latência e fraca escalabilidade [15]. No entanto, em função dos cenários da sua aplicação e dos requisitos impostos por estes, as suas mais-valias podem superar largamente as suas lacunas. De facto, os reduzidos custos de implementação e a sua flexibilidade são mais-valias destas redes, que muitas vezes levam a que sejam escolhidas em vez de outras alternativas [7]. Originalmente desenvolvidas para uso militar, as redes em malha sem fios são atualmente aplicadas num grande conjunto de áreas, tais como: saúde, comércio, indústria e meios rurais [16]. Estas podem ainda ser usadas para cobertura de grandes áreas no acesso à Internet em caso de desastres, incêndios ou terramotos [17]. A flexibilidade destas redes permite-lhes continuarem funcionais mesmo que falhe um dos nós, visto que estes não estão dependentes uns dos outros [6]. Uma característica importante destas redes é a inexistência de hierarquia, sendo que todos os nós participam ativamente no processo de transmissão de dados, ao contrário de outras topologias mais comuns. Isto torna a rede mais eficiente e tolerante a falhas [18]. Outras características importantes são os baixos custos de implementação, uma vez que é uma rede sem fios, e a sua fácil manutenção [19]. Mesmo considerando todas as suas mais-valias, estas redes apresentam ainda algumas desvantagens e desafios. A segurança é uma questão importante, uma vez que os dados passam por vários nós, podendo ser intercetados [20]. Muitas das aplicações das redes em malha podem ser móveis, o que significa que a infraestrutura física tem de ser montada para se adaptar às mudanças que ocorram [19].

A camada de aplicação tem ainda de ser desenvolvida para acomodar os vários tipos de dispositivos por onde a informação circula, uma vez que estes podem não suportar o formato em que os dados se encontram [19]. Outros problemas são levantados quanto à perda de pacotes na transmissão dos dados usando o *Transmission Control Protocol* (TCP), que se agrava nas redes sem fios. Esta lacuna pode ser colmatada pela utilização conjunta do *User Datagram Protcol* (UDP), do *Real Time Protocol* (RTP) e do *Rate Control Protocol* (RCP) nas redes sem fios [19].

As redes em malha podem ser configuradas com (Figura 1) ou sem nó raiz (Figura 2). Tal significa que, caso um nó raiz seja definido, todos os restantes nós se irão tentar conectar a esse nó. Por outro lado, a não existência de um nó raiz significa que os nós se irão conectar arbitrariamente. A utilização de um nó raiz pode ser útil para acelerar o processo da criação da malha, especialmente quando novos nós se conectam à rede ou quando nós existentes se desconectam e a rede tem de ser reconfigurada [21].

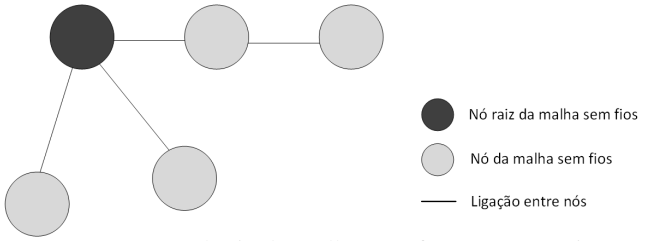

Figura 1. Topologia de malha sem fios (com nó raiz).

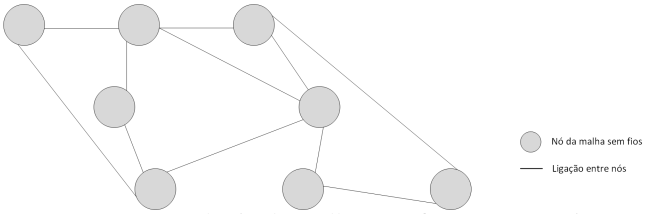

Figura 2. Topologia de malha sem fios (sem nó raiz).

No contexto deste trabalho, para implementar a rede em malha foi utilizada a biblioteca painlessMesh [13]. Esta biblioteca está disponível para o ambiente de desenvolvimento Arduino [22].Esta biblioteca usa as bibliotecas nativas dos dispositivos ESP8266 ou ESP32, aos quais se destina, prescindindo das bibliotecas WiFi do Arduino. A painlessMesh encarrega-se de fazer toda a configuração e gestão da rede em malha ao conectar nós com o mesmo *service set identifier* (SSID). Usando esta biblioteca é possível criar uma rede adhoc entre dispositivos ESP8266 ou ESP32 sem necessidade da existência de um router. O tamanho máximo da malha está limitado pela memória dos próprios dispositivos. Estas redes são implementadas na camada 3 do modelo OSI (camada de rede). Não sendo redes TCP/IP, os nós são identificados por um endereço de 32 bits gerado a partir do seu endereço MAC. Cada nó atualiza a topologia existente a cada 3 segundos, informando os seus vizinhos. As mensagens, trocadas em broadcast ou unicast, usam o formato JavaScript object notation (JSON). Isto facilita a compreensão por parte do programador e a sua integração em aplicações web, podendo, no entanto, diminuir o desempenho da rede [21]. A biblioteca painlessMesh permite configurar um nó como nó raiz da rede e tem uma metodologia própria para fazer a gestão desse nó (Figura 3) [21]. Para o efeito, faz uso de métodos nativos que devem ser invocados pelo programador. Este terá de ter sempre presente que cada rede em malha apenas poderá ter um único nó raiz e que os demais nós têm de ser configurados para que saibam da existência desse nó raiz. Os restantes nós, quando em operação, deverão, sempre que possível, conectar-se preferencialmente ao nó raiz. Os métodos mencionados serão detalhados na secção 3.2 do documento. Para o desenvolvimento deste trabalho foi construída uma rede em malha sem fios, gerida através da biblioteca painlessMesh, usando cinco nós sensores e um nó raiz. Na próxima secção será detalhado o cenário usado na validação da rede e todos os seus detalhes de implementação.

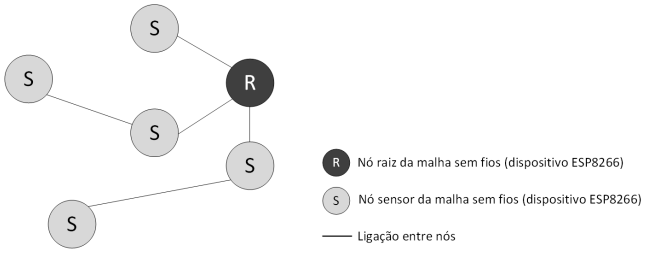

Figura 3. Exemplo de topologia da painlessMesh (com nó raiz).

3. Cenário de Aplicação

O transporte de fruta em fresco é uma das fases do pós-colheita onde pode verificar-se o risco de quebra da cadeia de frio, o que torna esta etapa crítica no processo de conservação. Para assegurar a qualidade da fruta e minimizar riscos da sua deterioração, são utilizadas caixas refrigeradas, transportadas em câmaras frigoríficas de camiões de mercadorias. Mantendo o foco na garantia da qualidade desta fruta, torna-se importante monitorizar indicadores como a temperatura e a humidade ao longo do processo de armazenagem e transporte. O cenário experimental construído para este trabalho pretendeu representar fielmente a galera de um camião de transporte de mercadorias. No caso, considerou-se o equivalente à galera de um camião com capacidade de transporte de 11 Europaletes [23]. Este tem as medidas de 5,9 metros (comprimento) x 2,4 metros (largura) x 2,4 metros (altura), o que se traduz num volume de 35 m3. A Figura 4 ilustra a representação do cenário descrito.

Para dar suporte à monitorização dos parâmetros de temperatura e humidade no cenário descrito, este trabalho focou-se na implementação de uma rede em malha sem fios, que possibilite a comunicação entre cinco nós sensores e um nó raiz, para envio em tempo real dos dados recolhidos. Os cinco nós sensores foram alimentados por baterias enquanto o nó raiz

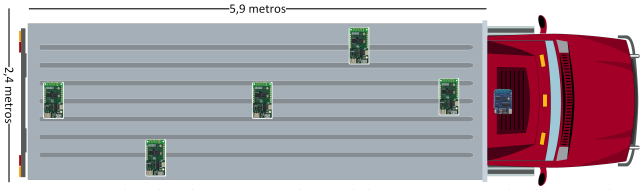

Figura 4. Distribuição dos dispositivos no cenário ilustrativo do camião.

dispôs de alimentação permanente. A rede em malha sem fios foi construída usando dispositivos ESP8226 [12] com sensores de temperatura e humidade integrados. Estes dispositivos serão parte integrante de caixas de fruta inteligentes, desenvolvidas no âmbito do projeto PrunusPós, e com a finalidade de aumentar a vida útil da fruta. Durante o trajeto de transporte, algumas caixas de fruta poderão ser removidas dos camiões para serem entregues aos seus clientes finais. Esta situação leva a que a rede em malha seja altamente volátil em termos de topologia e inclusivamente no número de nós presentes na rede. Seguidamente será detalhada a construção da rede em malha sem fios e os desenvolvimentos de suporte ao seu funcionamento.

3.1 Construção da Rede em Malha Sem Fios

A solução proposta passou pela criação de uma rede em malha sem fios usando cinco nós Wemos D1 Mini Pro [24] e um Wemos D1 Mini [25] (Figura 5), tendo sido este último usado como nó raiz. Todos estes nós integram o módulo ESP8266. As conclusões apresentadas em [11] permitiram demonstrar a viabilidade da utilização destes dispositivos e da biblioteca painlessMesh na criação de redes em malha sem fios.

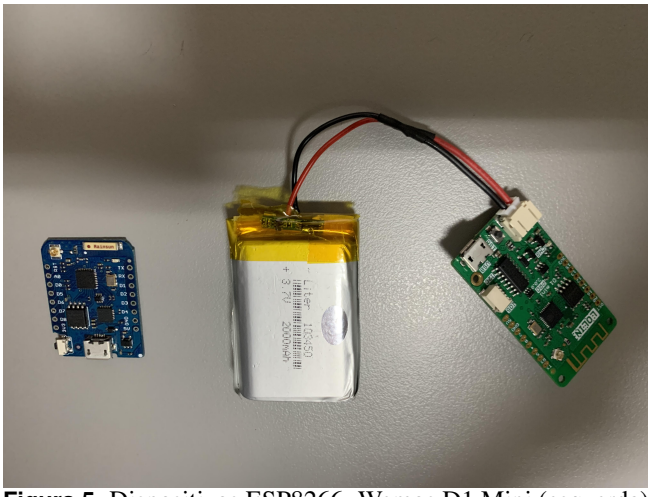

Figura 5. Dispositivos ESP8266. Wemos D1 Mini (esquerda) e Wemos D1 Mini Pro (direita).

Os Wemos D1 Mini Pro tratam-se de dispositivos com 16 MB de memória flash e que podem operar com uma frequência de relógio de 80 ou 160 Mhz [24]. O Wemos D1 Mini dispõe de 4 MB de memória flash e iguais frequências de relógio [25]. Todos os dispositivos transmitem através de uma antena wi-fi embutida, que trabalha na frequência de 2.4 Ghz [12]. Ambos os dispositivos podem ser programados através do Arduino IDE [26]. Entre as várias configurações que este IDE disponibiliza, é possível alterar o tamanho máximo dos pacotes (do inglês, *Maximum Segment Size*, MSS), através da opção *IwIP variant* [12]. O MSS corresponde ao tamanho da pilha de software de rede interna, que é gerida às custas da memória RAM ou FLASH dos dispositivos. O programador pode optar por utilizar segmentos de 536 bytes ou 1460 bytes.

Um dos requisitos impostos pelo projeto PrunusPós era a impossibilidade de recarregar as baterias dos nós regularmente. Desta forma, as soluções desenvolvidas no âmbito do projeto deveriam permitir a maior autonomia possível dos nós. Para dar resposta a esse requisito, foi necessário minimizar todos os consumos desnecessários de energia e maximizar a duração das suas baterias. Os dispositivos ESP8266 contam com três modos de adormecimento que lhes permitem poupar energia: *Modem-sleep*, *Light-sleep* e *Deep-sleep*. A Tabela 1 apresenta a comparação entre estes três modos.

Após análise dos dados apresentados na tabela, foi considerada a utilização do modo *Deep-sleep* no desenvolvimento da solução apresentada. Este modo é aquele que apresenta um menor impacto nos gastos energéticos, desligando todo o poder computacional do nó. No entanto, esta solução implica outro tipo de constrangimentos ao nível da formação da rede em malha sem fios. Estes constrangimentos e todos os detalhes do desenvolvimento desta solução serão discutidos na secção seguinte.

3.2 Desenvolvimento da Solução

Nos testes realizados ao longo do desenvolvimento deste trabalho foi possível confirmar que, a cada adormecimento de um nó, este se desliga da rede em malha. Tal significa que, a cada novo despertar, este nó terá de se reconectar à rede. Por este motivo, foi necessário analisar todas as situações que deveriam ser consideradas na solução a criar. A Tabela 2 descreve os vários desafios e respetivas hipóteses consideradas nesta análise.

Cada despertar do nó, bem como o tempo que este se encontra ativo, são também fatores críticos que condicionam a sua autonomia. Assim, foram objetivos fulcrais do trabalho desenvolvido, reduzir o número de vezes que cada nó tem de despertar e o tempo de atividade de cada nó, sem comprometer a taxa de entrega das leituras obtidas. No desenvolvimento dos algoritmos necessários ao funcionamento da rede em malha, foram utilizados vários métodos nativos da biblioteca painlessMesh. O nó raiz da malha foi definido através do método *setRoot()*. A utilização deste método requer que os demais nós conheçam a existência desse nó raiz na rede. Tal foi conseguido com recurso ao método *containsRoot()*. O método *isConnected()* é uma função auxiliar que permite validar que um nó se encontra diretamente conectado, ou que conhece um caminho até determinado nó. Este método foi também usado sempre que se pretendeu validar que um nó que despertou tinha forma de transmitir uma mensagem para

Tabela 1. Diferenças nos três modos de *sleep*, adaptado de Espressif [27].

		Modem-sleep	*Light-sleep*	*Deep-sleep*
Wi-Fi		Desligado	Desligado	Desligado
Relógio do Sistema		Ligado	Ligado	Ligado
Relógio de tempo real (RTC, *Real-Time Clock*)		Ligado	Ligado	Ligado
CPU		Ligado	Pendente	Desligado
Corrente do substrato		15 mA	0.4 mA	$20\mu A$
Corrente média	DTM[1] = 1	16.2 mA	1.8 mA	-
	DTM = 3	15.4 mA	0.9 mA	
	DTM = 10	15.2 mA	0.55 mA	

Tabela 2. Desafios que podem levantar constrangimentos da rede em malha sem fios.

Desafios	Hipóteses
Alguns nós podem não alcançar o nó raiz ou a rede em malha sem fios (não enviam a informação)	O que fazem com a informação recolhida? • Descartam-na? • Guardam-na na memória? • Voltam a tentar enviar? Apenas a informação da leitura atual? Ou também a que estava na memória? Desligam? • Quando voltam a ligar?
Ligam-se vários sensores ao mesmo tempo, não se conseguindo emparelhar com o nó raiz sem que haja um nó 'intermédio' no seu caminho	No limite, para a área em causa, todos os nós conseguem aceder diretamente ao nó raiz
Ligam-se vários sensores em simultâneo, sobrecarregando a rede	Criar assimetria no despertar, através da criação de um período de dormência aleatório
O tempo de atividade do nó pode não ser suficiente para formar/juntar-se a uma rede em malha e transmitir a informação	Fazer testes preliminares para determinar o tempo mínimo necessário para realizar essas tarefas
Ligam-se vários nós fora da proximidade geográfica mais imediata do nó raiz, podendo eventualmente formar ilhas	Fazer uso de métodos nativos para indicar que existe um nó raiz na rede Mesh, definindo o seu id, para forçar o encontro com o nó raiz
Os nós conseguem conectar e enviar a informação, mas não recebem confirmação da receção (ACK)	Voltam a enviar? Assumem que a mensagem foi recebida? No trabalho anterior [11], os autores observaram taxas de entrega suficientemente elevadas para o payload que será transmitido, sem necessidade de mecanismos de confirmação (ACK). Adormecem? Por quanto tempo?
O nó conecta, mas falha no envio da mensagem	Hipótese idêntica à anterior

o nó raiz. Em caso afirmativo, o envio das mensagens foi realizado via *unicast* através do método *sendSingle()*, sempre dirigido para o nó raiz da rede.

Para dar resposta a todos os desafios identificados, foram desenvolvidos os respetivos firmwares tanto para os nós sensores da rede como para o nós raiz. Na Figura 6 são apresentados os fluxogramas correspondentes aos firmwares desenvolvidos. Na avaliação da solução desenvolvida foi considerada a utilização de quatro modelos diferentes: modelo *DeepSleep* aleatório, modelo *DeepSleep* com duração definida, modelo ACK e modelo sem ACK. A distinção entre os modelos de *DeepSleep* residiu na duração do período de inatividade dos nós sensores, que é definida pelo parâmetro *SleepTime*. No modelo *DeepSleep* aleatório, a duração do *DeepSleep*, parâmetro *SleepTime*, é aleatória nos casos de insucesso na conexão à rede ou no envio das leituras. No modelo *DeepSleep* definido, a duração do *DeepSleep* será sempre constante, conforme valor introduzido pelo programador. No modelo ACK verifica-se a existência de uma mensagem de confirmação, pelo nó raiz, da receção das leituras enviadas pelos nós sensores. Por outro lado, no modelo sem ACK o nó raiz não envia uma mensagem de confirmação. A Figura 7 demonstra as diferenças entre os modelos com mensagem de confirmação (ACK) e sem mensagem de confirmação (sem ACK). Quando cruzados, estes modelos deram origem a quatro sub-modelos: Modelo ACK com *DeepSleep* aleatório, Modelo ACK com *DeepSleep* com duração definida, Modelo sem ACK com *DeepSleep* aleatório e Modelo sem ACK com *DeepSleep* com duração definida.

Considerando que um dos principais objetivos dos firmwares desenvolvidos foi aumentar a autonomia dos dispositivos, estes devem despender o menor tempo possível ligados. No

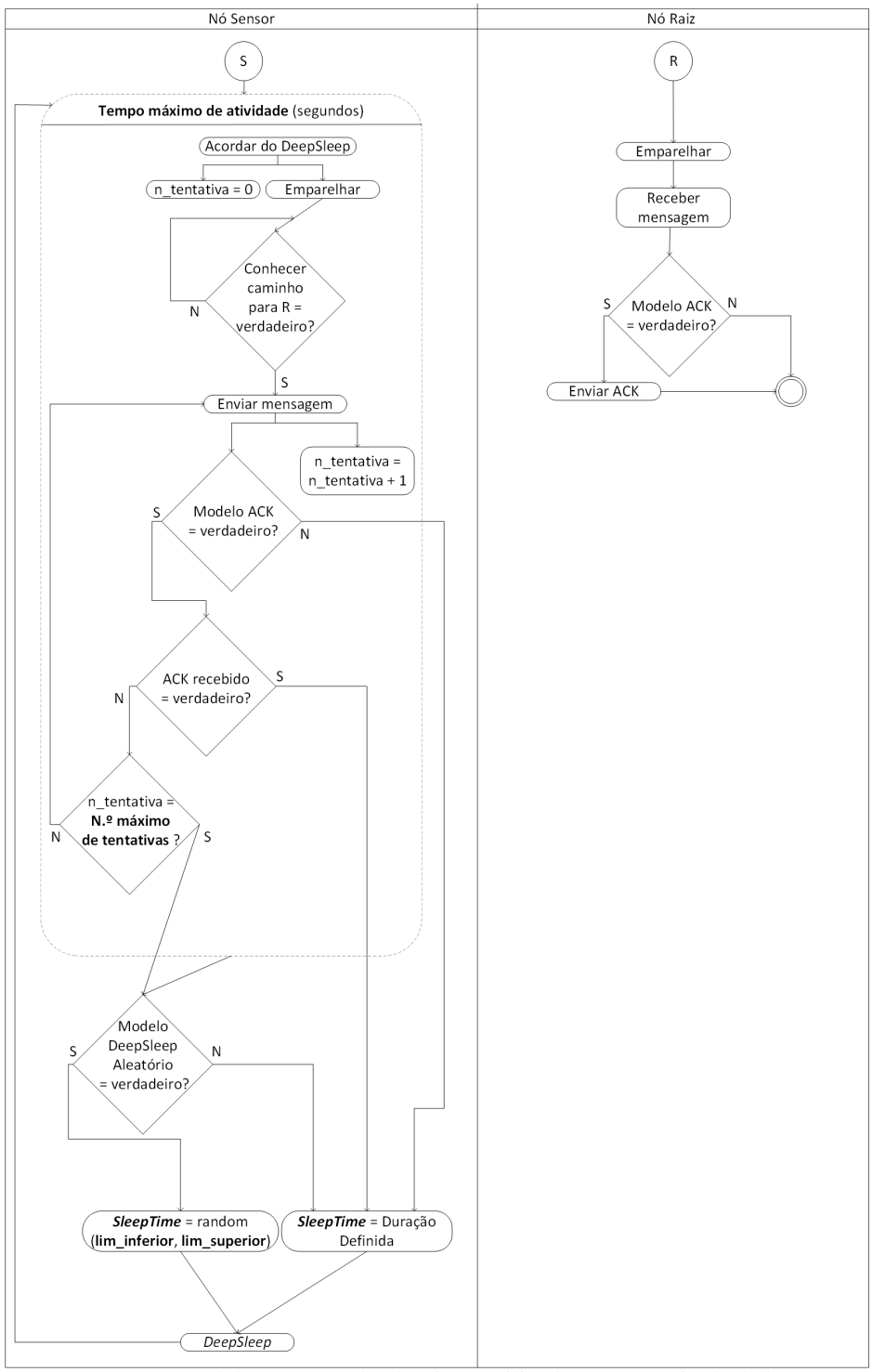

Figura 6. Modelação UML do algoritmo utilizado para o envio de dados.

entanto, este período (tempo máximo de atividade) deverá ser suficiente para que seja possível aos nós conectarem-se à rede e enviar a mensagem. Decorrido este tempo, o nó sensor (S) irá entrar no estado *DeepSleep*, mesmo que não tenha tido sucesso na entrega da mensagem. O processo de entrada no *DeepSleep* pode, no entanto, ser antecipado caso se verifique um evento que tenha um resultado bem-sucedido, por exemplo, receção de uma mensagem de confirmação (quando requerida), ou envio da mensagem quando o modelo implementado não exige a receção da mensagem de confirmação. Quando desperta do estado de *DeepSleep*, o dispositivo deve emparelhar com os restantes nós da malha, tendo sempre pre-

sente que existe um nó raiz. Após emparelhar, o dispositivo verifica se conhece um caminho até esse nó. Em caso afirmativo, irá enviar uma mensagem com a leitura da temperatura e da humidade. A solução proposta não guarda as leituras na memória dos dispositivos, sendo que estes dados serão descartados sempre que o sensor entre em *DeepSleep*. De cada vez que o nó sensor desperta, este deverá realizar uma nova leitura e apenas essa será enviada.

No modelo com mensagem de confirmação (ACK), o nó raiz deverá responder ao nó sensor com uma mensagem ACK de confirmação sempre que receba uma leitura. Para maximizar a garantia de receção dessa confirmação, foi definido um número máximo de tentativas de envio da mensagem que um nó sensor deverá realizar antes de entrar no estado *DeepSleep*. As mensagens foram retransmitidas quando o nó sensor não recebeu a mensagem de confirmação. A Figura 8 apresenta a metodologia do reenvio da mensagem até o número máximo de mensagens ter sido atingido. Quando recebe a confirmação, o nó sensor entra de imediato no estado de *DeepSleep*. Nos modelos em que não foi requerida a mensagem ACK de confirmação, o nó sensor entra no estado de *DeepSleep* assim que envia a mensagem.

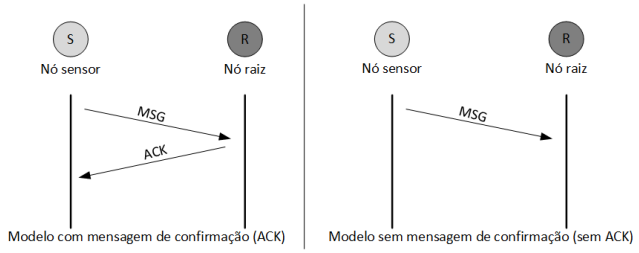

Figura 7. Envio de mensagem nos modelos com e sem confirmação de mensagem (ACK).

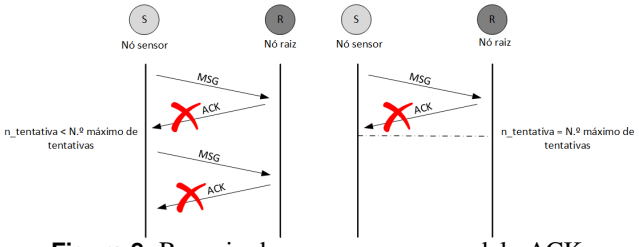

Figura 8. Reenvio de mensagens no modelo ACK.

No desenvolvimento da solução, um problema identificado foi o sincronismo dos nós, uma vez que, caso estes não despertassem ao mesmo tempo, poderia significar que nós sensores mais distantes do nó raiz poderiam não ter cobertura de rede para enviar os seus dados. Para determinar a relevância do sincronismo no despertar de todos os nós, foi utilizado um input, aqui designado por *SleepTime* (duração do *DeepSleep*), como parâmetro do método que invoca o *DeepSleep*. Este representa a duração de tempo que cada nó sensor irá estar inativo, em *DeepSleep*. Ao longo das experiências realizadas para avaliação e validação da solução proposta, descritas na secção seguinte, foram usados para este parâmetro um valor fixo de 300 segundos ou um valor aleatório dentro de um intervalo de tempo entre os 30 e os 300 segundos.

4. Avaliação de Desempenho

Esta secção é dedicada à apresentação dos resultados obtidos na avaliação de desempenho da rede em malha sem fios, com recurso aos dispositivos ESP8266 e à biblioteca painlessMesh, no cenário apresentado. Para a avaliação de desempenho foram considerados os quatro modelos anteriormente descritos. Dentro destes, fez-se o cruzamento entre existir mensagem de confirmação ou não (ACK/sem ACK) e a duração do *DeepSleep*, o *SleepTime*, em caso de insucesso na entrega da mensagem. O *SleepTime* pode ter um valor definido, no caso cinco minutos, ou ser aleatório dentro de um intervalo especificado. No caso aleatório, o limite inferior do intervalo foi definido em trinta segundos, para evitar que o nó desperte em intervalos de tempo demasiado curtos, com isso desperdiçando bateria; o limite superior foi estabelecido em cinco minutos, o que corresponde ao caso de sucesso de envio da mensagem ou da sua confirmação, quando aplicável. Cada um destes modelos foi avaliado através da recolha de dados durante uma hora ininterrupta, através do *Serial Monitor* do Arduino IDE. Neste período, foi registado se cada nó sensor conseguiu conectar quando despertou do estado *DeepSleep* e se este enviou mensagem para o nó raiz; foi também registado o número da tentativa de envio da mensagem, se a mensagem foi recebida no nó raiz e se o nó sensor recebeu a mensagem de confirmação (ACK) do nó raiz, quando aplicável. Com base nos resultados obtidos, foi contabilizada a média dos rácios de todos os nós sensores, para cada modelo que fora estabelecido.

4.1 Modelos Implementados e Métricas de Desempenho

A programação do firmware foi realizada com recurso ao Arduino IDE [26]. Em função dos resultados obtidos em [11], que evidenciaram que não há benefícios evidentes em utilizar pacotes com um MSS de 1460 bytes em detrimento de um MSS de 536 bytes, os testes foram realizados com um MSS de 536 bytes. As restantes configurações, no que diz respeito à velocidade do CPU, utilizam os valores por omissão de 80 Mhz. A Tabela 3 explicita os quatro modelos que foram implementados. O trabalho pretendeu avaliar se existem vantagens em introduzir uma mensagem de confirmação sempre que o nó raiz recebe uma mensagem ou se esta é desnecessária. O trabalho visou também avaliar se os nós sensores devem despertar de forma síncrona ou se, em caso de insucesso nas suas tarefas, por exemplo, tempo máximo de atividade excedido, falha no envio da mensagem ou na receção da sua confirmação, os nós sensores devem despertar após períodos aleatórios, de forma assíncrona.

Nos dois modelos em que é esperada uma mensagem de confirmação (ACK), o nó sensor enviou até duas vezes a mensagem, caso não tivesse recebido confirmação da primeira.

Tabela 3. Modelos implementados na avaliação de desempenho.

	ACK Duração Aleatória	ACK Duração Definida	Sem ACK Duração Aleatória	Sem ACK Duração Definida
Tempo máximo de atividade (segundos)	15	15	15	15
Modelo ACK (mensagem de confirmação)	Sim	Sim	Não	Não
Número máximo de tentativas de envio de mensagem	2	2	1	1
Duração do *DeepSleep* (*SleepTime*) em caso de sucesso (segundos)	300	300	300	300
Duração do *DeepSleep* (*SleepTime*) em caso de insucesso (segundos)	Aleatório [30, 300]	300	Aleatório [30, 300]	300

Nos dois modelos em que não é requerida mensagem de confirmação (sem ACK), os nós sensores apenas enviaram a mensagem uma vez. A duração do *DeepSleep* foi estabelecida em 300 segundos (5 minutos) nos casos em que o *SleepTime* foi definido como um período fixo. Nos modelos de duração aleatória, o *SleepTime* apenas foi fixado em 300 segundos quando a execução das suas tarefas foi bem-sucedida. Quando foi registado um evento de insucesso nas suas tarefas, o nó sensor entrou no estado de *DeepSleep* por um período aleatório compreendido entre 30 segundos e 300 segundos. O limite inferior foi estabelecido para 30 segundos para minimizar a quantidade de vezes que o nó desperta. O limite superior é igual à duração do *DeepSleep* no modelo não aleatório. Em todos os modelos foi definido um tempo máximo de atividade de 15 segundos. Este valor foi assim estipulado visto ser um tempo adequado para o sensor enviar a mensagem, preservando o máximo de bateria possível. De modo a avaliar a performance da rede, foram consideradas quatro métricas de desempenho: taxa de conectividade, taxa de envio após conectividade e as taxas de entrega das primeira e segunda tentativas, esta última apenas quando aplicável. A taxa de conectividade pretende traduzir a percentagem de vezes que um nó desperta do estado de *DeepSleep* e se consegue conectar com outro nó. A taxa de envio após conectividade representa a percentagem de vezes que um nó consegue enviar uma mensagem, sempre que tenha conectado a outro nó. A taxa de entrega na primeira tentativa consiste no rácio de mensagens recebidas no nó raiz face ao total de mensagens enviadas pelo nó sensor na primeira tentativa. A taxa de entrega na segunda tentativa corresponde ao rácio de mensagens recebidas no nó raiz do total de mensagens que foram enviadas pelo nó sensor numa segunda tentativa. A subsecção seguinte apresenta os resultados obtidos e faz a sua análise.

4.2 Análise de Resultados

Para avaliar o desempenho da rede em malha sem fios, foram conduzidos os modelos e configurações atrás identificados. Seguem os resultados obtidos e a suas análise e discussão.

4.2.1 Taxa de Conectividade

A primeira métrica avaliada foi a taxa de conectividade. A Figura 9 permite comparar o desempenho dos quatro modelos ao nível desta métrica.

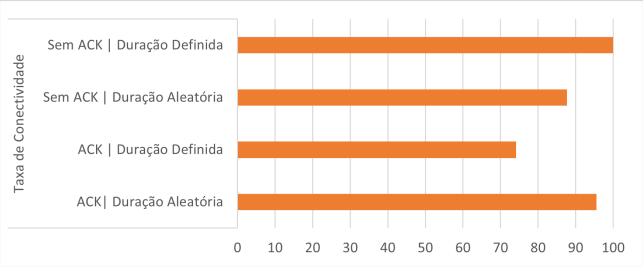

Figura 9. Taxa de conectividade (%).

O modelo que apresentou melhor desempenho foi o que não recorreu a mensagens de confirmação de receção e cujo período de *DeepSleep* (*SleepTime*) foi mantido constante (sem ACK, com duração definida), com uma taxa de conetividade de 100.00%. Os resultados sugerem que, se os sensores tiverem sido ligados em simultâneo, os seus períodos de adormecimento serão consistentes. Deste modo, foi observado sincronismo entre os nós sensores, o que poderia antecipar um maior congestionamento da rede nos períodos de despertar. Este fenómeno de congestão da rede foi registado, contudo, os dispositivos possuem mecanismos ao nível das exceções para mitigar estes eventos. Em caso de sobrecarga da rede no período de emparelhamento, os dispositivos podem lançar uma exceção, provocando o seu *reboot* espontâneo e, com isso, atrasando ligeiramente o seu reinício e consequentemente o seu sincronismo com todos os outros nós sensores. O segundo modelo com o melhor comportamento foi aquele que introduziu um período aleatório de *DeepSleep* quando foi requerida mensagem de confirmação (95.52%). Este resultado sugere que, nos casos em que é necessário garantir fiabilidade de entrega das mensagens, deverá ser conduzido um desnivelamento nos períodos em que os sensores despertam do *DeepSleep*. O valor contrasta com os 74.19% de taxa de conectividade quando foi igualmente requerida confirmação da receção da mensagem, mas foi pretendido sincronizar o

despertar de todos os nós sensores. Estes valores sugerem que os nós sensores desperdiçaram bateria, como resultado de terem despertado e não terem conseguido emparelhar com qualquer outro dispositivo com o qual pudessem formar uma rede em malha.

4.2.2 Taxa de Envio Após Conectividade

A taxa de envio após conectividade pretende avaliar a fiabilidade da conectividade entre os nós sensores e o nó raiz. De acordo com o algoritmo desenvolvido, cada nó sensor apenas enviou uma mensagem para o nó raiz caso estes estivessem diretamente conectados ou se existisse um caminho entre eles. A taxa de envio após conectividade não pretende avaliar se a mensagem foi efetivamente recebida, apenas se esta partiu do nó sensor com destino ao nó raiz. A Figura 10 apresenta os resultados desta métrica.

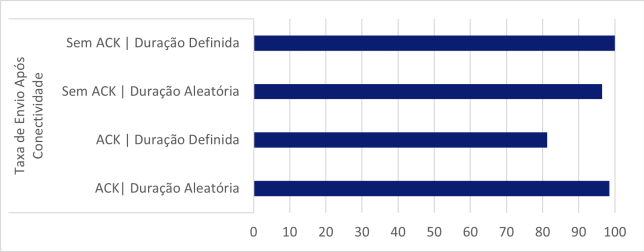

Figura 10. Taxa de envio após conectividade (%).

Os resultados evidenciam que o modelo em que não existiu confirmação da receção da mensagem e em que os nós sensores entraram em *DeepSleep* por um período definido (sem ACK, duração definida) apresentou a melhor performance na execução da tarefa de envio da mensagem (100.00%, i.e., sempre que conectaram com a malha). No entanto, a assimetria dos resultados entre o modelo com os melhores resultados e aquele em que o período de inatividade era igualmente definido, mas no qual existiu mensagem de confirmação manteve-se. Neste modelo com confirmação e *SleepTime* definido (ACK, *SleepTime* de duração definida), apenas em 81.25% das vezes a mensagem foi enviada pelos nós sensores. Não parece haver diferenças significativas entre os dois modelos que utilizaram um período aleatório de *SleepTime* (96.49%, no caso de não existir mensagem de confirmação e 98.44%, quando existiu mensagem de confirmação). Alguns fenómenos que podem justificar o facto de, sabendo que os nós sensores conseguiram conectar-se, existir uma disparidade tão significativa entre o modelo em que foi requerida mensagem de confirmação e a duração de *DeepSleep* foi constante (ACK, duração definida) e os outros três modelos são: perda de conectividade no caminho para o nó raiz, uma vez que o nó que intermediava a sua ligação entrou em período de *DeepSleep* (Figura 11), ou a formação de ilhas sem conectividade com o nó raiz (Figura 12). Foram também registadas situações em que o nó sensor consumiu a quase totalidade do seu tempo de atividade em emparelhamento com a rede. Deste modo, o nó sensor já não dispôs de tempo suficiente para enviar a mensagem até ao nó raiz.

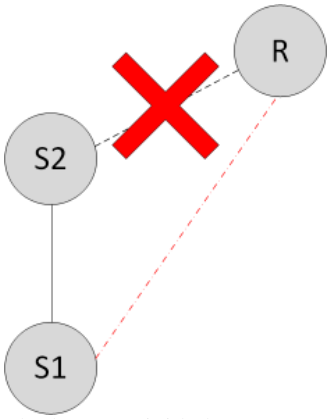

Figura 11. Perda de conectividade com um nó intermédio.

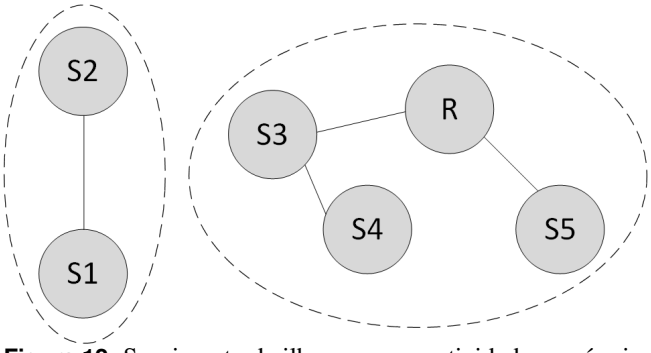

Figura 12. Surgimento de ilhas sem conectividade ao nó raiz.

4.2.3 Taxa de Entrega da Primeira Tentativa

A taxa de entrega da primeira tentativa pretende avaliar o sucesso da entrega das mensagens enviadas pelos nós sensores para o nó raiz, no primeiro envio.

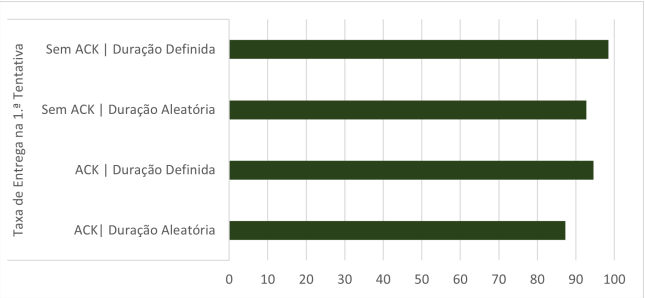

Figura 13. Taxa de entrega da 1.ª tentativa (%).

A Figura 13 evidencia que o modelo sem envio de confirmação (ACK) e com duração de *DeepSleep* definida de forma homogénea para todos os nós (sem ACK, duração definida) apresenta os melhores resultados na entrega da primeira tentativa do envio da mensagem (98.44%). Este modelo foi seguido de perto pelo modelo em que foi requerida uma mensagem de confirmação da receção por parte do nó raiz, mantendo a duração do DeepSleep constante (ACK, duração definida) (94.59%). Apenas o modelo com confirmação e duração de *DeepSleep* aleatório (ACK, duração aleatória) obteve um registo abaixo dos 90% (87.30%).

4.2.4 Taxa de Entrega da Segunda Tentativa

A taxa de entrega da segunda tentativa visou apurar se, nos modelos em que existiu o envio de mensagem de confirmação, foi útil retransmitir a mensagem numa perspetiva de rentabilização do despertar do nó sensor.

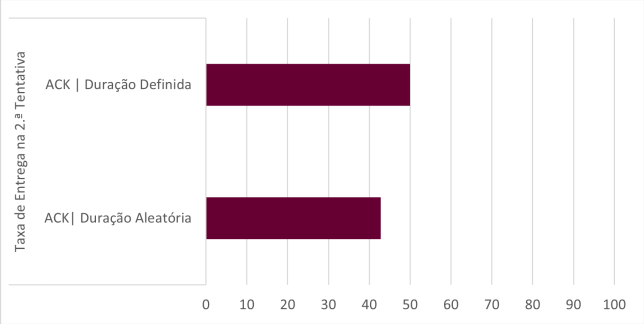

Figura 14. Taxa de entrega da 2.ª tentativa (%).

Os dados apresentados na Figura 14 permitem concluir que não houve qualquer vantagem em enviar a mensagem uma segunda vez, pois a sua taxa de entrega revelou-se insuficiente; no melhor caso, a taxa de entrega da segunda tentativa não ultrapassou os 50.00%. A introdução de uma segunda tentativa apenas acrescentou esforço computacional aos sensores e contribuiu para reduzir a autonomia da bateria, sem qualquer benefício evidente.

4.2.5 Sumário dos Resultados

A Tabela 4 pretende sintetizar os resultados obtidos nesta avaliação de desempenho e facilitar a sua comparação pelo leitor. Desta se conclui que o modelo sem envio de confirmação e com duração de *DeepSleep* definida como parâmetro de entrada (sem ACK, duração definida) teve sempre o melhor comportamento.

Conclusões

As redes em malha sem fios são uma tecnologia com a capacidade de mudar o paradigma da agricultura, dotando-a de instrumentos inteligentes. Este artigo apresentou uma avaliação de quatro modelos de redes em malha que pretendem ser aplicados no âmbito do projeto PrunusPós para o envio de mensagens com os valores da temperatura e da humidade entre cinco nós sensores e um nó raiz. Estes serão integrados em câmaras ou caixas de transporte frigoríficas em toda a cadeia logística do pós-colheita, nomeadamente no transporte.

A implementação recorreu à utilização de dispositivos ESP8266 e à biblioteca painlessMesh, que, quando combinados, formam uma rede sem fios em malha. Foram avaliadas quatro métricas de desempenho, designadamente a taxa de conectividade, a taxa de envio após conectividade e as taxas de entrega das primeira e segunda tentativas.

O algoritmo produzido e avaliado neste artigo revelou-se profícuo no compromisso entre autonomia e desempenho, minimizando o número de vezes que cada nó sensor desperta, bem como o seu tempo útil de atividade. Deixa ainda espaço para que o programador possa introduzir parâmetros que melhor se adequem às suas necessidades de recolha de dados, nomeadamente o tempo de atividade e a duração do *DeepSleep*.

Os dados apontam para que, aquando da utilização da biblioteca painlessMesh e dos dispositivos ESP8266, a introdução de uma mensagem de confirmação de receção das leituras não trouxe benefícios. Esta conduziu, pelo contrário, a piores resultados, dado o *overhead* adicional que criou na rede. Esta ilação é consistente com o trabalho anterior dos autores [11]. Aquele trabalho concluíra que a taxa de entrega de uma mensagem por segundo num cenário de cinco nós era suficientemente alta para, no contexto deste trabalho, não ser requerida confirmação. Porém, os mesmos autores assinalaram que a taxa de entrega se degradaria substancialmente com o aumento do número de mensagens a serem transmitidas em simultâneo.

Os resultados sugerem que, para cenários de implementação da painlessMesh que exijam a confirmação da receção dos dados, os tempos de inatividade de cada nó devem ser programados dentro de períodos aleatórios, pois a simultaneidade de nós a despertar pode provocar constrangimentos na rede, reduzindo, desde logo, a taxa de conectividade dos nós sensores. Deste modo, é assegurado que os nós sensores não despertam todos em simultâneo e, desde que estejam no perímetro de cobertura do nó raiz, a sua taxa de conectividade sai claramente beneficiada.

Os resultados obtidos complementam o trabalho anterior dos mesmos autores [11] e recomendam a realização dos ensaios dos algoritmos desenvolvidos no contexto real de conservação de fruta. Será necessário ter presente que a existência dos contentores e a água que compõe os frutos podem constituir barreiras adicionais na comunicação. Será necessário avaliar qual o seu impacto no desempenho da rede. Também o comportamento das baterias no frio deverá ser investigado. No campo da biblioteca painlessMesh, deverão ser estudados mecanismos que possam vir a enriquecer a segurança das transmissões à medida que a sua implementação se democratize no contexto da Internet das Coisas.

Contribuições dos autores

Tiago Costa, Luís Santos: investigação, metodologia, análise formal, validação, preparação de rascunho de redacção original.

João M. L. P. Caldeira, Vasco N. G. J. Soares: análise formal, validação, revisão-escrita e edição, supervisão.

Pedro D. Gaspar: revisão-escrita e edição, supervisão.

Tabela 4. Quadro sumário das métricas de desempenho.

	ACK Duração Aleatória	ACK Duração Definida	Sem ACK Duração Aleatória	Sem ACK Duração Definida
Taxa de Conectividade (%)	95.52	74.19	87.69	100.00
Taxa de Envio Após-Conectividade (%)	98.44	81.25	96.49	100.00
Taxa de Entrega da 1.ª Tentativa (%)	87.30	94.59	92.73	98.44
Taxa de Entrega da 2.ª Tentativa (%)	42.82	50.00	N/A	N/A

Referências

[1] MANDAL, N. C.; UDDIN, G. An empirical study of iot security aspects at sentence-level in developer textual discussions. *Information and Software Technology*, United Kingdom, v. 150, p. 106970, October 2022.

[2] FURSTENAU, L. B. et al. Internet of things: Conceptual network structure, main challenges and future directions. *Digital Communications and Networks*, China, p. 1–16, May 2022.

[3] KAKHI, K. et al. The internet of medical things and artificial intelligence: trends, challenges, and opportunities. *Biocybernetics and Biomedical Engineering*, Poland, v. 42, n. 3, p. 749–771, July 2022.

[4] KAGAN, C. R. et al. Special report: The internet of things for precision agriculture (iot4ag). *Computers and Electronics in Agriculture*, Netherlands, v. 196, p. 106742, May 2022.

[5] TAN, S. W.; LEE, S. C.; CHAN, C. L. Clonal-selection-based minimum-interference channel assignment algorithms for multiradio wireless mesh networks. In: *Bio-Inspired Computation in Telecommunications*. Boston, USA: Morgan Kaufmann, 2015. p. 287–321.

[6] PIECHOWIAK, M. et al. Comparative analysis of routing protocols for wireless mesh networks. In: *2016 10th International Symposium on Communication Systems, Networks and Digital Signal Processing (CSNDSP)*. USA: IEEE, 2016. p. 1–5.

[7] LIU, Y. et al. Wireless mesh networks in iot networks. In: *2017 International Workshop on Electromagnetics: Applications and Student Innovation Competition*. USA: IEEE, 2017. p. 183–185.

[8] SHAO, S. et al. A random switching traffic scheduling algorithm for data collection in wireless mesh network. In: *The 16th Asia-Pacific Network Operations and Management Symposium*. USA: IEEE, 2014. p. 1–4.

[9] SIMÕES, M. P. et al. Prunuspós—otimização de processos de armazenamento, conservação em frio, embalamento ativo e/ou inteligente, e rastreabilidade da qualidade alimentar no pós-colheita de produtos frutícolas. In: *Grupos Operacionais de Fruticultura no período 2018-2022*. Portugal: COTHN - Centro Operativo e tecnológico Hortofrutícola Nacional, 2021. p. 404–471.

[10] GO Prunus Pós. Disponível em: <https://prunospos.webnode.pt/>. Acesso em: 2 de abril de 2022.

[11] SANTOS, L. et al. Performance assessment of esp8266 wireless mesh networks. *Information*, Switzerland, v. 13, n. 5, p. 1–15, April 2022.

[12] GROKHOTKOV, I. *ESP8266 Arduino Core Documentation*. Disponível em: <https://readthedocs.org/projects/arduino-esp8266/downloads/pdf/latest/>. Acesso em: 2 de abril de 2022.

[13] PAINLESSMESH Technical Documentation. Disponível em: <https://gitlab.com/painlessMesh/painlessMesh/-/wikis/home>. Acesso em: 13 de junho de 2022.

[14] KUMAR, K. A.; HEGDE, S. Multicasting in wireless mesh networks: Challenges and opportunities. In: *2009 International Conference on Information Management and Engineering*. USA: IEEE, 2009. p. 514–518.

[15] PANDI, S.; WUNDERLICH, S.; FITZEK, F. H. P. Reliable low latency wireless mesh networks — from myth to reality. In: *2018 15th IEEE Annual Consumer Communications & Networking Conference (CCNC)*. USA: IEEE, 2018. p. 1–2.

[16] KÖBEL, C.; GARCÍA, W. B.; HABERMANN, J. A survey on wireless mesh network applications in rural areas and emerging countries. In: *2013 IEEE Global Humanitarian Technology Conference (GHTC)*. USA: IEEE, 2013. p. 389–394.

[17] AKYILDIZ, I. F.; WANG, X.; WANG, W. Wireless mesh networks: a survey. *Computer Networks*, Netherlands, v. 47, p. 445–487, March 2005.

[18] CILFONE, A. et al. Wireless mesh networking: An iot-oriented perspective survey on relevant technologies. *Future Internet*, Switzerland, v. 11, n. 4, p. 1–35, April 2019.

[19] SHAHDAD, S. Y.; SABAHATH, A.; PARVEEZ, R. Architecture, issues and challenges of wireless mesh network. In: *2016 International Conference on Communication and Signal Processing (ICCSP)*. USA: IEEE, 2016. p. 557–560.

[20] SICHITIU, M. L. Wireless mesh networks: Opportunities and challenges. In: . [S.l.: s.n.], 2005.

[21] GITLAB. *painlessmesh::Mesh< T > Class Template Reference*. Disponível em:

<https://painlessmesh.gitlab.io/painlessMesh/classpainlessmesh_1_1Mesh.html>. Acesso em: 2 de abril de 2022.

[22] ARDUINO. *Arduino IDE 2 Tutorials*. Disponível em: <https://docs.arduino.cc/software/ide-v2>. Acesso em: 13 de junho de 2022.

[23] SOLUTIONS, R. L. *Tipologia de Camiões*. Disponível em: <https://www.rangel.com/pt/infohub/tipologia-de-camioes/>. Acesso em: 16 de abril de 2022.

[24] WEMOS. *D1 mini Pro*. Disponível em: <hhttps://www.wemos.cc/en/latest/d1/d1_mini_pro.html>. Acesso em: 2 de abril de 2022.

[25] WEMOS. *LOLIN D1 mini*. Disponível em: <https://www.wemos.cc/en/latest/d1/d1_mini.html>. Acesso em: 2 de abril de 2022.

[26] ARDUINO. *Arduino | Software*. Disponível em: <https://www.arduino.cc/en/software>. Acesso em: 2 de abril de 2022.

[27] INC, E. *ESP8266 - Low Power Solutions*. Disponível em: <https://www.espressif.com/sites/default/files/9b-esp8266-low_power_solutions_en_0.pdf>. Acesso em: 2 de abril de 2022.